Linguagem de Programação Python

Listas

Professor: Ritomar Torquato

07 Listas

Objetivos: Conhecer o tipo de dados Lista;

Introdução

Lista é um tipo de variável que permite o armazenamento de vários valores acessados por um índice.

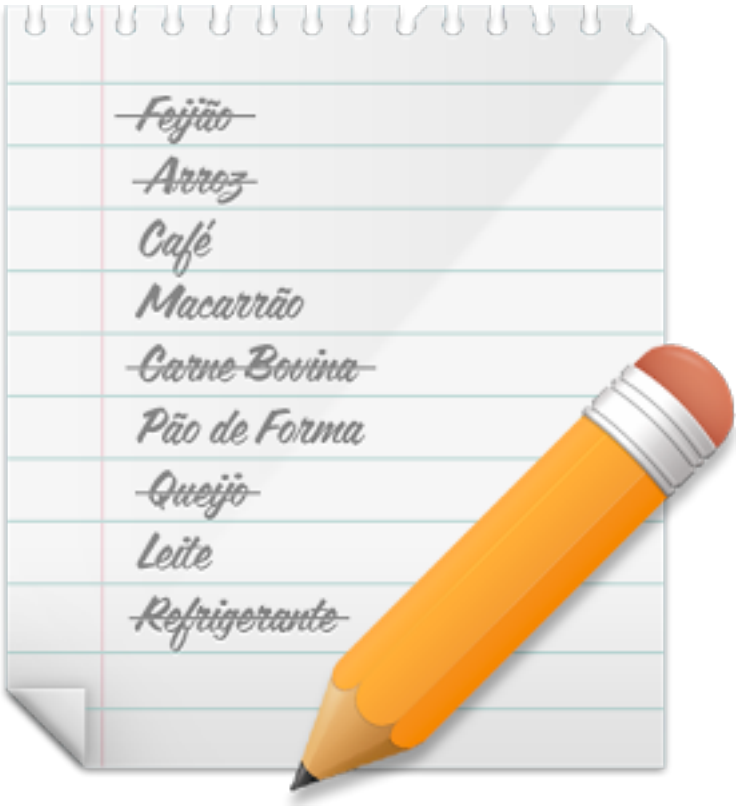
O tamanho de uma lista é a quantidade de elementos que ela contém.

Introdução

Podemos imaginar uma lista como um edifício. O térreo é o andar zero, o primeiro andar é o primeiro item da lista e assim por diante.

Em cada andar podemos armazenar "coisas".

Introdução



Em uma lista de compras com nove itens teremos os índices variando de 0 a 8.

Se chamarmos nossa lista de compras por C, teremos C[0] feijão, C[1] arroz, C[2] café, continuando até o item C[8] refrigerante.

Listas são flexíveis e podem crescer ou diminuir com o tempo.

Introdução

Em Python temos:

Uma Lista Vazia.

```
>>> L = []
```

Cria uma lista chamada L que está vazia (zero itens).
Os colchetes ([ ]) indicam que L é uma lista.

Introdução

Em Python temos:

Uma Lista com 3 elementos.

```
>>> Z = [ 15, 8, 9 ]
```

A lista Z foi criada com 3 elementos.
Dizemos que o tamanho de Z é 3.

Introdução

Em Python temos:

Uma Lista com 3 elementos.

```
>>> Z = [ 15, 8, 9 ]
>>> Z[0]
15
>>> Z[0] = 7
>>> Z[0]
7
>>> Z
[7, 8, 9]
```

Podemos mudar o valor de um elementos com uma atribuição.

Introdução

Um aluno tem 5 notas e desejamos calcular sua média aritmética.

```
notas = [6, 7, 5, 8, 9]
soma = 0
i = 0
while i < 5:
    soma += notas[i]
    i += 1
print("Média: %5.2f" % (soma/5))
```

Todas as notas foram armazenadas na lista, utilizando um índice para armazenar cada valor.

Prática

- Calcule a média aritmética de um aluno que possui 7 notas. Use a estrutura de repetição mais adequada.

Trabalhando com índices

Um programa ler 5 números e depois solicita que o usuário escolha um número para mostrar.

```
números = [0, 0, 0, 0, 0]

for i in range(5):
    números[i] = int(input("Número %d: " % (i + 1)))

while True:
    escolhido = int(input("Que posição você quer mostrar (0 para sair): "))
    if escolhido == 0:
        break
     print("Você escolheu o número %d" % números[escolhido-1])
```

A

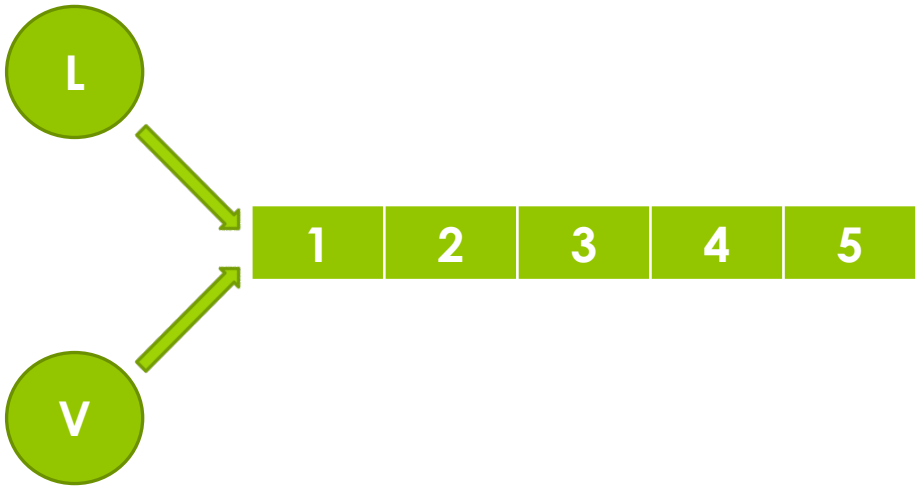
B

Em A adicionamos 1 para imprimir de 1 a 5, não é natural começar a contar de zero. Em B fizemos a operação inversa.

Cópia e fatiamento de listas

Cópia de Listas:

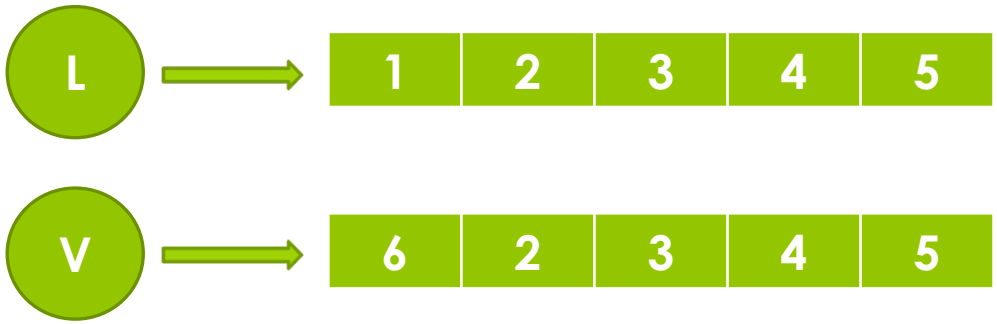
```
>>> L = [ 1, 2, 3, 4, 5]
>>> V = L
>>> L
[1, 2, 3, 4, 5]
>>> V[0] = 6
>>> V
[6, 2, 3, 4, 5]
>>> L
[6, 2, 3, 4, 5]
```



Cópia e fatiamento de listas

Cópia de Listas:

```
>>> L = [ 1, 2, 3, 4, 5]
>>> V = L[:] # Nova cópia de L
>>> V[0] = 6
>>> L
[1, 2, 3, 4, 5]
>>> V
[6, 2, 3, 4, 5]
```



Cópia e fatiamento de listas

Fatiamento de Listas:

```
>>> L = [1, 2, 3, 4, 5]
>>> L[ 0 : 5]
[1, 2, 3, 4, 5]
>>> L[ : 5]
[1, 2, 3, 4, 5]
>>> L[ : -1]
[1, 2, 3, 4]
>>> L[ 1 : 3]
[2, 3]
```

```
>>> L[ 1 : 4]
[2, 3, 4]
>>> L[ 3: ]
[4, 5]
>>> L[ : 3]
[1, 2, 3]
>>> L[-1]
5
>>> L[-2]
4
```

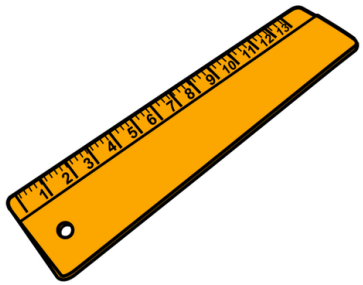
Um índice negativo conta a partir de último elemento. Se L[0] é sempre o primeiro elemento L[-1] é o último, L[-2] o penúltimo... .

Tamanho de listas

A função `len()` retorna a quantidade de elementos em uma lista.

Tamanho da Lista

```
>>> L = [ 12, 9, 5]
>>> len(L)
3
>>> V = []
>>> len(V)
0
```



Repetição sem `len()`

```
>>> L = [ 1, 2, 3]
>>> i = 0
>>> while i < 3:
        print(L[i])
        i += 1
```

Repetição com `len()`

```
>>> L = [ 1, 2, 3]
>>> i= 0
>>> while i < len(L):
        print(L[i])
        i += 1
```

Adição de elementos

Podemos adicionar elementos durante a execução.

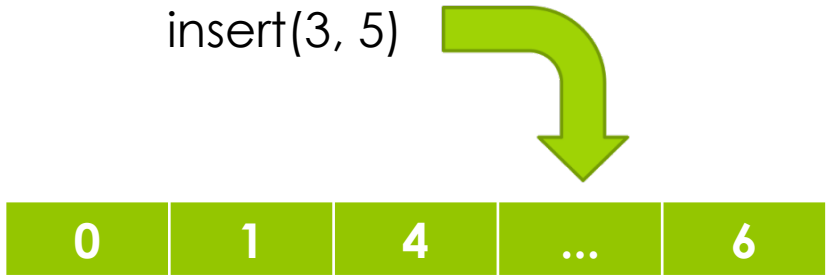
```
L = []

while True:
    n = int(input("Digite um número (0 para sair): "))
    if n == 0:
        break
    L.append(n)

for i in L:
    print(i)
```

Append adiciona um elemento no final da lista.

Adição de elementos

Podemos adicionar elementos durante a execução.

```
>>> L = [ 0, 1, 4, 6]
>>> L.insert(3, 5)
>>> L
[0, 1, 4, 5, 6]
>>> L.insert(2, 3)
>>> L
[0, 1, 3, 4, 5, 6]
>>> L.insert(2, 2)
>>> L
[0, 1, 2, 3, 4, 5, 6]
```



Insert adiciona um elemento na posição desejada da lista.

Adição de elementos

Adição como concatenação de listas.

```
>>> L = []
>>> L = L + [1]
>>> L
[1]
>>> L += [2]
>>> L
[1, 2]
>>> L += [3, 4, 5]
>>> L
[1, 2, 3, 4, 5]
```

```
>>> L = []
>>> L.extend([1])
>>> L
[1]
>>> L.extend([2])
>>> L
[1, 2]
>>> L.extend([3, 4, 5])
>>> L
[1, 2, 3, 4, 5]
```

Internamente o interpretador executa o método extend que adiciona os elementos de uma lista a outra.

Adição de elementos

Adição de elementos do tipo Lista.

```
>>> L = ["a"]
>>> L.append(["b"])
>>> L.append(["c", "d"])
>>> L
['a', ['b'], ['c', 'd']]
>>> len(L)
3
```

```
>>> L[1]
['b']
>>> L[2]
['c', 'd']
>>> len(L[2])
2
>>> L[2][1]
'd'
```

Dessa forma é possível usar estruturas de dados como matrizes, árvores e registros.

Remoção de elementos

Usamos `del` para remover elementos da Lista.

```
>>> L = [ "a", "b", "c"]
>>> del L[1]
>>> L
['a', 'c']
>>> del L[0]
>>> L
['c']
```

Remover fatias

```
>>> L = list(range(101))
>>> del L[1:99]
>>> L
[0, 99, 100]
```

Aplicações

Lendo e imprimindo uma lista de compras.

```
compras = []

while True:
    produto = input("Produto ('fim' para terminar): ")
    if produto.upper() == "FIM":
        break
    compras.append(produto)

for p in compras:
    print(p)
```